Project Report - Data Science II

Data Science II (MECH-B-5-MLDS-MLDS2-ILV)

Spam Filter

Bachelor program - Mechatronik, Design und Innovation

5th semester

Lecturer: Daniel McGuiness

Group: BA-MECH-22-MRV

Authors: Jonas Pluder

January 5, 2025

Contents

1 Introduction

The primary objective of this project is to design and implement a spam classifier to effectively detect unsolicited and harmful emails. This involves processing email datasets, extracting relevant features, and building machine learning models to achieve high precision and recall. The project explores classification techniques and natural language processing (NLP) methods using Python.

1.1 SCOPE

The spam filter developed here can be integrated into email systems to improve the user experience by automatically filtering spam emails into a separate folder. This project also provides hands-on experience with machine learning (ML) pipelines and data filtering techniques.

1.2 TOOLS AND LIBRARIES

The implementation is done using Python, with libraries such as:

- NLTK: For natural language processing.
- Scikit-learn: For machine learning models and evaluation metrics.
- Pandas and NumPy: For data manipulation and analysis.
- Regex: For preprocessing text data.
- BeautifulSoup: For parsing HTML content in emails.

2 Dataset Overview

2.1 SOURCE

The dataset is sourced from local directories containing labeled spam and ham emails. These emails simulate real-world email traffic.

2.2 STRUCTURE

The dataset includes text files with email content. Each file represents an individual email and is labeled as either spam or ham.

3 Methodology

3.1 DATA PREPARATION

1. **Data Loading**:
    - Emails are loaded from specified directories for spam and ham.
    - Each email is parsed using the `email.parser` module to extract content.
2. **Data Cleaning**:
    - Removed email headers and metadata.

- Converted email content to lowercase.
- Removed punctuation and special characters.
- Replaced URLs with the placeholder `URL` and numbers with `NUMBER`.
- Extracted text content from HTML emails using BeautifulSoup.

3. **Feature Extraction**:
   - Transformed emails into feature vectors using a Bag-of-Words (BoW) approach.
   - Each email is represented as a sparse vector indicating the presence or absence of words.
4. **Data Splitting**:
   - Split the dataset into training (80%) and testing (20%) sets.

3.2 MODEL SELECTION

The following classifiers were implemented and evaluated:

- **Naïve Bayes**: Baseline model for text classification.
- **Logistic Regression**: For binary classification with linear decision boundaries.
- **Random Forest**: Ensemble model leveraging multiple decision trees.
- **Voting Classifier**: Combines predictions from Naïve Bayes, Logistic Regression, and Random Forest using soft voting.

3.3 EVALUATION METRICS

The models were evaluated based on:

- **Precision**: Proportion of true positives among predicted positives.
- **Recall**: Proportion of true positives among actual positives.
- **F1-Score**: Harmonic mean of precision and recall.
- **Accuracy**: Overall performance of the model.

4 Results

PERFORMANCE METRICS

The final model, a Voting Classifier combining Naïve Bayes, Logistic Regression, and Random Forest, achieved the following results on the test set:

Metric	**Value**
Precision	0.99
Recall	0.96
F1-Score	0.97
Accuracy	0.98

4.1 DETAILED CLASSIFICATION REPORT

```
              precision    recall  f1-score   support

           0       0.98      1.00      0.99       497
           1       0.99      0.92      0.95       114

    accuracy                           0.98       611
   macro avg       0.99      0.96      0.97       611
weighted avg       0.98      0.98      0.98       611
```

4.2 OBSERVATIONS

- The Voting Classifier demonstrated robust performance, achieving high precision, recall, and F1-scores.
- The model's ensemble nature contributed to its ability to generalize well across the dataset.

5 Discussion

5.1 CHALLENGES

- Parsing and cleaning HTML-heavy email content.
- Ensuring a balance between precision and recall to minimize false positives and false negatives.

5.2 DECISIONS

- A Voting Classifier was chosen to leverage the strengths of multiple models by combining their predictions. This approach improves overall performance by aggregating the unique strengths of each classifier—Naïve Bayes excels in probabilistic reasoning, Logistic Regression provides robust performance for linearly separable data, and Random Forest contributes by handling complex, nonlinear relationships. By using soft voting, the Voting Classifier assigns weights to predictions based on class probabilities, enhancing its ability to make more accurate decisions.
- Precision was prioritized in this project to avoid the significant inconvenience caused by false positives, where legitimate emails are flagged as spam. This decision reflects the need to preserve user trust and ensure critical communications are not missed. However, this focus comes with a trade-off: slightly reduced recall, meaning some spam emails may evade detection. The balance was deemed acceptable given the context of the application, where false negatives are less damaging than false positives.

5.3 FUTURE IMPROVEMENTS

- Incorporate Support Vector Machines (SVMs) to explore their impact on performance.
- Experiment with advanced feature extraction techniques, such as TF-IDF.

6 Conclusion

This project successfully developed a spam classifier using an ensemble of Naïve Bayes, Logistic Regression, and Random Forest. The Voting Classifier achieved high precision and recall, demonstrating its effectiveness in real-world scenarios. Future enhancements could further improve its robustness and applicability.